REMARKS

This Amendment is filed in response to the Final Office Action of January 3, 2005 within a two-month period on March 3, 2005. The Applicants thank the Examiner for courtesy extended to the undersigned during the telephone interview on January 8, 2005. The Applicants also thank the Examiner 's supervisor, Kakali Chaki, for the courtesy extended to the undersigned during a telephone interview on March 2, 2005.

Claims 1-20 are pending after entry of the present Amendment.

Rejections under 35 U.S.C. § 102(b):

Claims 1-6 and 8-20 are rejected under 35 U.S.C. § 102(b) as being anticipated by Aho et al. (Hereinafter "Aho"), "Compilers: Principles, Techniques, and Tools." Applicants respectfully traverse.

In one embodiment of the present invention, code generation is simultaneously optimized for at least two target machines. A rule of instruction scheduling is abstracted from each of the at least two target machines. A hypothetical machine is generated based on the abstracted rule of instruction scheduling from each of the at least two target machines. The hypothetical machine is then targeted for generating optimized code. *See page 11, lines 9-26.*

Having incorporated the rule of instruction scheduling from at least two target machines, according to one embodiment, the hypothetical machine is targeted for generating compatible and functionally acceptable code for multiple target machines. *See page 13, lines 8-11.* For example, the hypothetical machine is capable of addressing latency, register pressure, and blocking versus pipelining constrains of the target machines. *See page 18, lines 11-18.*

In contrast, Aho teaches the general techniques of building compilers. Aho explains that a compiler is a program that reads a program written in one language, the source language, and translates it into an equivalent program in another language, the target language. *See page 1, second paragraph.*

Aho also explains that there are two parts of compilation: analysis and synthesis. The analysis part breaks up the source program into constituent pieces and creates an intermediate representation of the source program. The synthesis part constructs the desired target program from the intermediate representation. *See page 2, fourth paragraph.* The target program is then used to generate code in the desired target language. *See Figure 1.1 and Figure 8.1.*

The intermediate representation can be considered as a program for an abstract machine. *See page 12, first paragraph under section "Intermediate Code Generation."* The Examiner equates the abstract machine of Aho as the hypothetical machine of the present invention. *See page 2 of the Final Office Action.* Although the name "abstract machine" appears to be similar to the name "hypothetical machine", they are different. The hypothetical machine, according to one embodiment of the present invention, is used for generating optimized code, whereas the abstract machine of Aho is used to construct the target program. *See page 12, first paragraph under section "Intermediate Code Generation".* The abstract machine is an intermediate representation; it is not yet capable of generating the target code. *See Figure 8.1*

Furthermore, the hypothetical machine is generated based on the rule of instruction scheduling for each of at least two target machines. In contrast, the abstract machine of Aho is an intermediate representation that is constructed from the constituent pieces of the source code. Moreover, the target program of Aho, constructed from the abstract machine, is also a

descendant of the source program. Thus, neither the abstract machine, nor the target program of Aho is generated from the rule of instruction scheduling of at least two target machines.

Aho further discusses the analysis-synthesis model of a compiler in which the front end, using syntax-directed methods, translates a source program into an intermediate representation from which the back end generates a target code from the target program. *See, Chapter 8, Intermediate Code Generation on page 463.* The Examiner equates the syntax-directed method of translating the source program into the intermediate representation as abstracting rule of instruction scheduling for each of at least two target machines. *See page 2 of the Final Office Action.* Translating the source program is not the same as abstracting rule of instruction scheduling from target machines. Because target machines, as they pertain to the present invention, are not the source; instead, they are the recipients of the translated source code. In effect, the source code is where the translation begins and the translated code is provided to the target machines at the end of the translation process. According to one embodiment of the present invention, rule of instruction scheduling is abstracted from the target machines to ensure that the optimized code is compatible and functionally acceptable code for multiple target machines as recipients of the translated source code. Accordingly, abstracting rule of instruction scheduling from the source code, as Aho is suggesting, would not ensure the translated code is compatible or function effectively for multiple target machines.

In a code optimization phase, attempts are made to improve the intermediate representation or code so that a faster-running machine code will result. *See page 14, first paragraph under "Code Optimization".* Optimization improves the run time of the target program. *See page 15, second paragraph.* The Examiner equates code optimization of Aho as the same as code generation optimized for at least two target machines according to one embodiment of the present invention. *See page 2 of the Final Office Action.*

The optimization in Aho involves improving the run time of the target program. Whereas, according to one embodiment of the present invention, optimization involves generating a hypothetical machine based on the abstracted rule of instruction scheduling from at least two target machines. Incorporating the rule of instruction scheduling from at least two target machines, the hypothetical machine is then able to generate compatible and functionally acceptable code for multiple target machines. The ability to generate compatible and functionally acceptable code for multiple target machines is not the same as improving run time for a target machine.

Based on the differences discussed above, Aho fails to teach each and every feature of independent claims 1, 8, 13, 19, and 20. Accordingly, Aho does not anticipate independent claims 1, 8, 13, 19, and 20. Dependent claims 2-6, 9-12, 14-18 drawing their respective dependencies from independent claims 1, 8, or 13 are similarly not anticipated by Aho for substantially the same reasons as discussed above and for the additional limitations that each dependent claims respectively recites.

Rejections under 35 U.S.C. § 103(a):

Claim 7 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Aho in view of "UltraSPARC-III: Designing Third-Generation 64-Bit Performance" (Hereinafter "III"), and Sun Micro System press release of May 1998 (Hereinafter "II'). Applicants respectfully traverse.

Assuming, *in arguendo*, that there is a suggestion or motivation to combine the references, a proposition that the Applicants would disagree, the combination of the references still does not teach each and every feature of dependent claim 7. That is, even if reference III and II disclose the features of dependent claim 7, the combination of Aho, III,

and II must disclose each and every feature of claim 7 and independent claim 1, because claim 7 draws its dependency from independent claim 1.

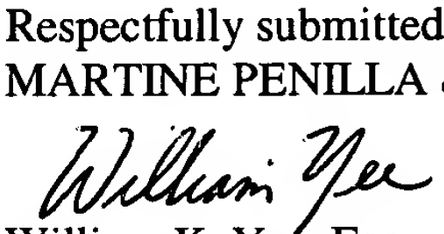
As discussed in the above section, Aho fails to each and every feature of independent claim 1. Since neither III nor II remedies the deficiency of Aho, the combination of Aho, III, and II fails to disclose each and every feature of claim 7, because claim 7 includes all the limitations of independent claim 1.

Since the combination of Aho, III, and II fails to disclose each and every feature of claim 7, the combination fails to render claim 7 obvious. Accordingly, dependent claim 7 is patentable.

Hence, after entry of the present Amendment, the application is now in a condition for allowance. A Notice of Allowance is therefore respectfully requested.

If the Examiner has any questions concerning the present Amendment, the Examiner is kindly requested to contact the undersigned at (408) 774-6911. If any other fees are due in connection with filing this Amendment, the Commissioner is also authorized to charge Deposit Account No. 50-0805 (Order No. SUNMP303). A duplicate copy of the transmittal is enclosed for this purpose.

Respectfully submitted,
MARTINE PENILLA & GENCARELLA, LLP



William K. Yee, Esq.
Reg. No. 54,943

710 Lakeway Drive, Suite 200
Sunnyvale, CA 94085
Telephone: (408) 774-6900, ext. 6911
Facsimile: (408) 749-6901
Customer No. 32291